17.04.2012

NWK (MI2)

Vorlesung 4

Inhalt

Netzwerk - Socket-Programmierung
Teil 1 - Grundlagen
Probleme bei heterogenen Netzen
Standardaustauschformate
Teil 2 - Laborübungen

Thema

Referenzen

Grundlagen - Standardaustauschformate

- Probleme bei Heterogenen Netzen
    - Datenrepräsentation
    - Datenaustauschformate

Standardaustauschformate: Binäre Formate; XDR und ASN
Textbasierende Austauschformate

Binäre Austauschformate: z.B.: ASN1

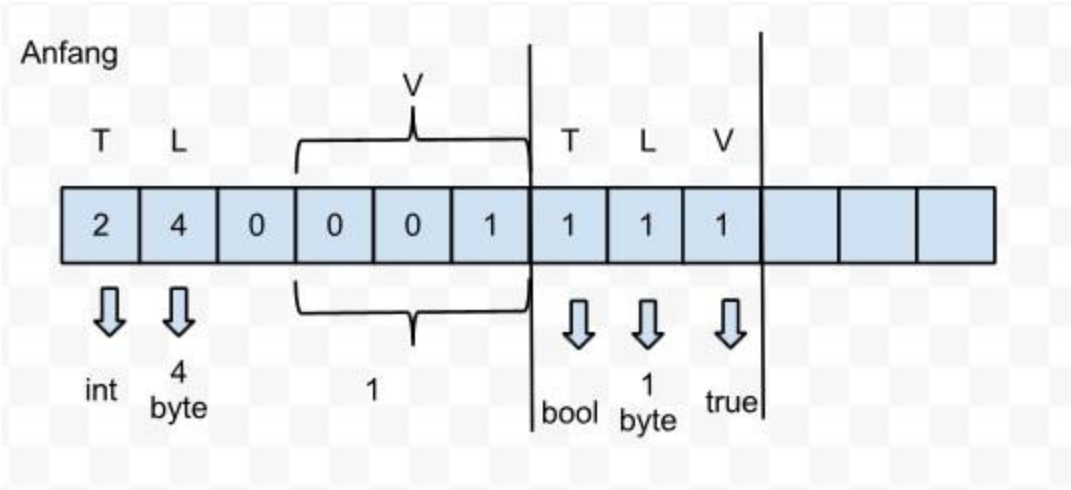
- Codierungsverfahren Type-Length-Value Encoding (TLV)

- Egal, in welchem Format die verschiedenen Datentypen gespeichert sind, Typen und Reihenfolge der ausgetauschten Daten sind Fall-abhängig (Problem - was kommt in einem konkreten Beispielfall als Dateityp an ?)

- Es reicht nicht, wenn ein Empfänger weiß, wie z-B. double-Werte kodiert sind; er muss auch wissen, ob als nächstes ein double-Wert ankommt.

Type-Length-Value Encoding (TLV)

- ein bewährtes Konzept für binäre Daten ist das “Type-Length-Value Encoding”. -



→ für Text-basierte Datenformate - werden in der Regel
Auszeichnungens-Sprachen (Mark-Up-Languages; z.B.:HTML) mit entsprechenden Mark-Ups/Tags verwendet und/oder entsprechen Felder im Protocol-Header (z.B.: http → ContentTyp, Content-Length, …. s.a. MIME-Types)

http Protocol Demo

XML- ist eine Meta-MarkUpLanguage - es sind zwar Syntaxregeln definiert ABER es ist ein Framework (definieren Sie sich eigene Tags)

Grundlagen - Schnittstelle vs. Protokoll

→ S/W-Schnittstelle = API (Applic. Programming I/F)

- Gesamtheit aller Beziehungen zwischen Modulen
- für andere Module bereitgestellte Definitionen, Methoden usw = “provided I/F”
  +
- von anderen Modulen benötigte Definitionen, Methoden usw. = “required I/F”

Praxis - Interface = provided Interface

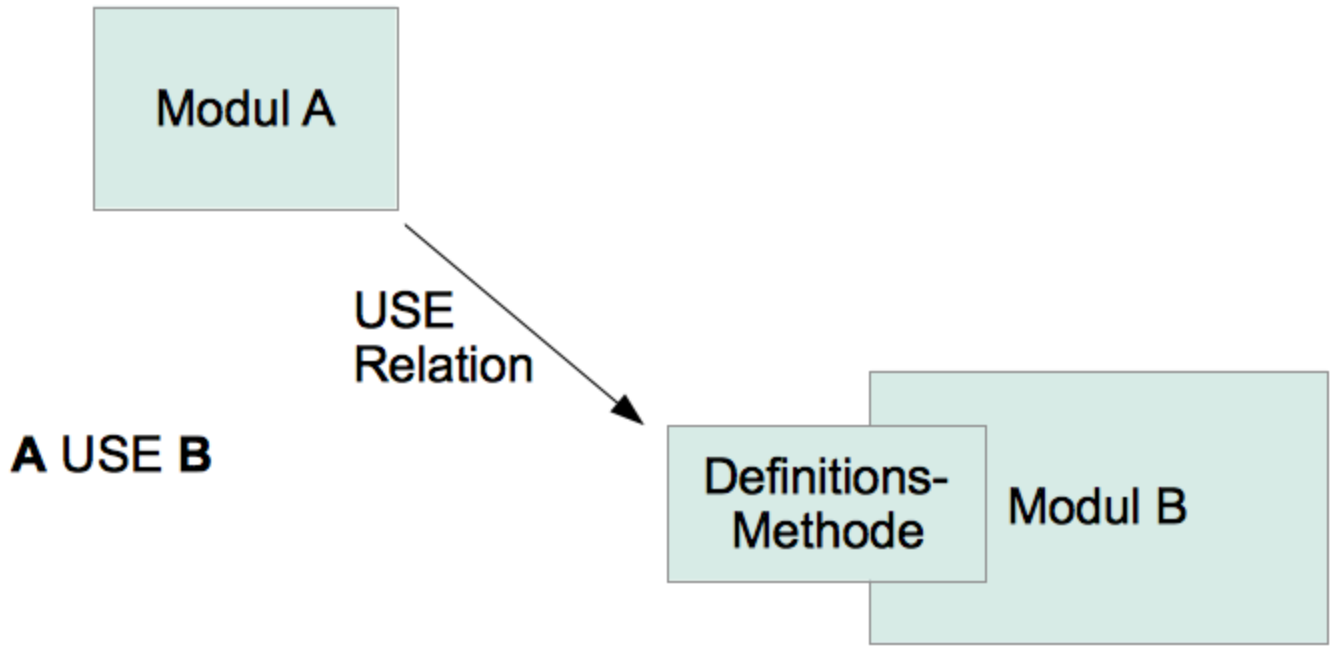
→ **Interfaces sind keine Klassen** ←

Erläuterung und Definition von Interfaces erfolgt später in GUI-Programmierung

- Beispiel für einseitige USE-Relation
Abb. 2)

Grundlagen - Protokoll

- Satz von Regeln/Vereinbarungen zwischen kommunizierenden Partnern (→ Peers) über:
  - Formate und Bedeutung der Daten
  - Aktivitäts- und Aufrufreihenfolge
  - Wortzeilen



  - Kontroll-Informationen werden i.d. Regel den Daten in einem “Header” vorangestellt

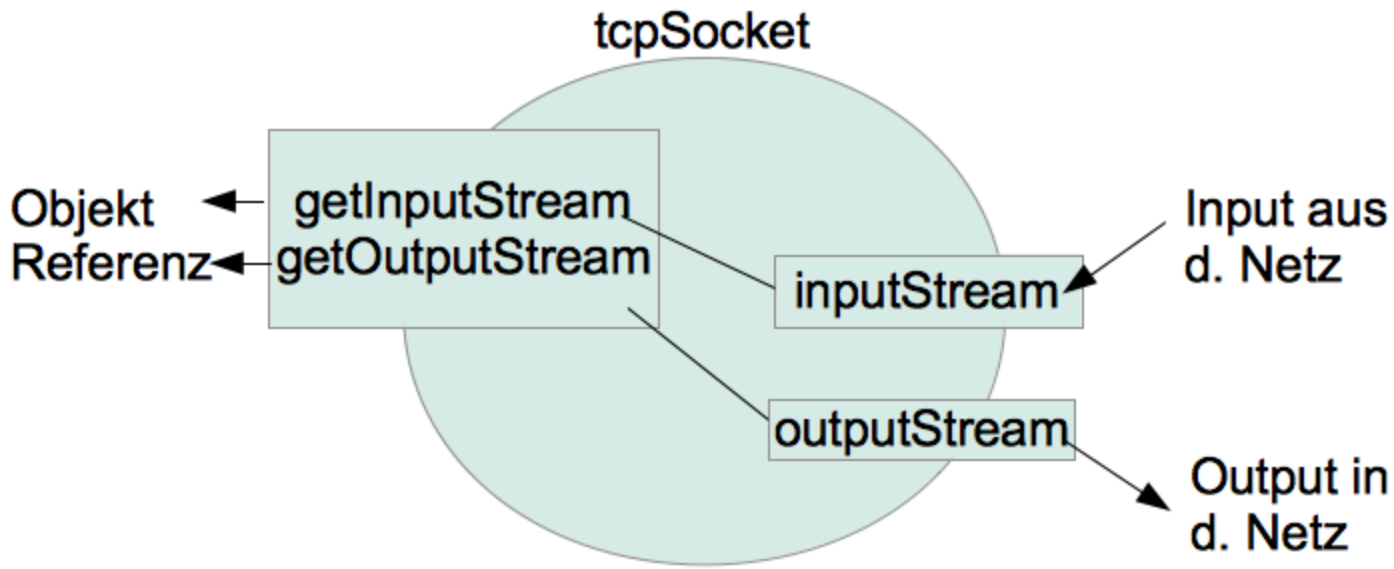
Beispiel für ein Protokoll:
“Ich klingel dich 3x an, wenn ich gut angekommen bin; 4x hat andere Bedeutung, ect. .”

Grundlagen - Interfaces - API vs. Protocol-based Interfaces
- Client/Server Interfaces können reine APIs oder Protokoll-basiert sein
(siehe PDF - Types of Remote Service Interface)

LABOR-HINWEISE

TCP - Socket-Programmierung

- Dateien sind I/O-Resources … wie Sockets/Streams (über das Netz)
- TCP-Sockets ermöglichen Byte-Stream-I/O über das Internet. (verbindungs-orientierte) Dazu besitzt jedes top-Socket-Objekt jeweils ein Input-Stream und ein OutputStream-Objekt. (Objekte entsprechen Subklassen) und die Methoden getInputStream() und getOutputStream() für den Zugriff auf diese Objekte:



- abb. 3 -

```
...new InputStreamReader(topSocket.getInputStream());
```

vergleiche File-I/O:

```
...new BufferedReader(new InputStreamReader(file));
```

TCP-Sockets werden in Java Unterschieden in :

- “einfache” Sockets - class Socket
    - können connect.Requests verschicken
    - und “Nutzdaten” übertragen

- Server-Sockets: - class ServerSocket
    - “lauschen” auf connect-Requests und nehmen diese Requests an …
    - erzeugen bei Bedarf “einfache” Sockets und verbinden sie mit dem jeweiligen Client für den Datentransport

Server: (textbasierte Kommunikation)
// Serversocket für connect-Requests erstellen

```
ServerSocket listenSocket= new ServerSocket(portNo);
//portNumber(int)
…
```
// Server wartet auf connect-Requests
// → dann: socket für die Client-Kommunikation:
```
Socket comSocket = listenSocket.accept();
```
// BufferedReader zum lesen (von Text) aus dem comSocket erzeugen:
```
BufferedReader bin;
bin = new BufferedReader(new
InputStreamReader(comSocket.getInputStream()));
…
comSocket.close();
listenSocket .close();
```

Client:
// Socket erzeugen, der connect-Request an angegebenen Server schickt:

IP/Punktnotation oder Domain-Name

|

V

```
Socket s = new Socket(serverHost, serverPort);
```

↑int

// erzeugt Exception, wenn nach Timeout keine Verbindung zustande kommt
```
BufferedWriter bout;
bout = new BufferedWriter(new OutputStreamWriter
(s.getOutputStream));
…
s.close();
```